AFHRL-TP-90-76

(2)

AIR FORCE

HUMAN RESOURCES

MODELING INDIVIDUAL DIFFERENCES IN PROGRAMMING SKILL ACQUISITION

Valerie J. Shute
Patrick C. Kyllonen

MANPOWER AND PERSONNEL DIVISION
Brooks Air Force Base, Texas 78235-5601

December 1990

Final Technical Paper for Period August 1988 - September 1990

Approved for public release; distribution is unlimited.

LABORATORY

AIR FORCE SYSTEMS COMMAND
BROOKS AIR FORCE BASE, TEXAS 78235-5601

NOTICE

When Government drawings, specifications, or other data are used for any purpose other than in connection with a definitely Government-related procurement, the United States Government incurs no responsibility or any obligation whatsoever. The fact that the Government may have formulated or in any way supplied the said drawings, specifications, or other data, is not to be regarded by implication, or otherwise in any manner construed, as licensing the holder, or any other person or corporation; or as conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

The Public Affairs Office has reviewed this paper, and it is releasable to the National Technical Information Service, where it will be available to the general public, including foreign nationals.

This paper has been reviewed and is approved for publication.

WILLIAM E. ALLEY, Technical Director
Manpower and Personnel Division

MICHAEL W. BIRDLEBOUGH, Colonel, USAF
Chief, Manpower and Personnel Division

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave blank)

2. REPORT DATE
December 1990

3. REPORT TYPE AND DATES COVERED
Final Paper - August 1988 - September 1990

4. TITLE AND SUBTITLE
Modeling Individual Differences in Programming Skill Acquisition

5. FUNDING NUMBERS
PE - 61102F
PR - 2313
TA - T1
WU - 44

6. AUTHOR(S)
Valerie J. Shute
Patrick C. Kyllonen

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Manpower and Personnel Division
Air Force Human Resources Laboratory
Brooks Air Force Base, Texas 78235-5601

8. PERFORMING ORGANIZATION REPORT NUMBER
AFHRL-TP-90-76

9. SPONSORING/MONITORING AGENCY NAMES(S) AND ADDRESS(ES)

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

This paper investigates individual differences in learning Pascal programming skills using an intelligent tutoring system for the learning criterion task. Data were collected from 260 subjects in the following categories: (a) incoming general and domain-specific knowledge; (b) cognitive ability measures (working-memory capacity and information processing speed); (c) learning process measures (declarative and procedural); and (d) transfer measures (retention, application, and generalization). Causal models of learning were tested linking incoming knowledge and abilities to learning processes and transfer. Findings showed that a large proportion (86%) of the learning criterion variance (transfer) was accounted for by just three factors: working-memory capacity, domain-specific knowledge, and declarative-learning efficiency. Certain learning behaviors were also investigated in relation to transfer. Some influenced transfer performance early in the tutor while others had more impact later on. Implications of these findings are discussed in relation to individual differences research as well as to intelligent tutoring system design issues.

14. SUBJECT TERMS

domain-specific knowledge	Pascal programming
individual differences	skill acquisition
intelligent tutoring system	transfer

15. NUMBER OF PAGES
34

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT
Unclassified

18. SECURITY CLASSIFICATION OF THIS PAGE
Unclassified

19. SECURITY CLASSIFICATION OF ABSTRACT
Unclassified

20. LIMITATION OF ABSTRACT
UL

AFHRL Tèchnical Paper 90-76 December 1990

MODELING INDIVIDUAL DIFFERENCES IN PROGRAMMING SKILL ACQUISITION

Valerie J. Shute
Patrick C. Kyllonen

MANPOWER AND PERSONNEL DIVISION
Brooks Air Force Base, Texas 78235-5601

Reviewed by

Dan J. Woltz
Cognitive Skills Assessment Branch

Submitted for publication by

Joseph L. Weeks, Chief
Cognitive Skills Assessment Branch

This publication is primarily a working paper. It is solely to document work performed.

SUMMARY

We investigated the development of computer programming skill in an intelligent tutoring system (ITS) by individual-differences analyses. Data were collected on general and domain-specific knowledge and abilities, tutor-specific learning processes (concept acquisition and skill development on the ITS), and a set of near, middle, and far transfer measures. Structural modeling analyses revealed that 74% of the variance in domain-specific declarative learning, a latent factor presumed to underlie all learning in the ITS, was accounted for by three factors: working-memory capacity, domain-specific knowledge, and general knowledge. Also, 10% of the variance in domain-specific procedural learning, a second latent factor orthogonal to associative learning presumed to underlie programming skill development per se, was accounted for by working-memory capacity. Finally, 86% of the variance in transfer, a third latent factor presumed to underlie performance on the transfer tests, was accounted for by a combination of domain-specific knowledge, domain-specific learning ability, and general working-memory capacity; there was no indication that the different transfer measures tapped different abilities. Through exploratory analyses of the relationship between learning behaviors and tutor and transfer performance, we found that good learners engaged in more exploratory activities (e.g., hint-asking) early in the tutor, but these activities were drastically reduced over time. We discuss contributions both to the individual differences in learning literature, and to the literature on optimizing ITS design.

PREFACE

We wish sincerely to thank Roy Chollman, Lisa Gawlick, Elena Selles, and Rich Walker for help in data collection. We are also indebted to Phil Ackerman, Wes Regian, Raymond Christal and especially Dan Woltz for suggestions during the conduct of this research. We would also like to thank the creative and industrious programmers: Tony Beauregard and Kym Costa.

The research reported in this paper was conducted by personnel of the Manpower and Personnel Division, Air Force Human Resources Laboratory, Brooks Air Force Base, Texas. The opinions expressed in this article are those of the authors and do not necessarily reflect those of the Air Force.

Correspondence concerning this article should be addressed to Valerie J. Shute, Air Force Human Resources Laboratory, Cognitive Skills Assessment Branch, Brooks Air Force Base, Texas 78235-5601.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

MODELING INDIVIDUAL DIFFERENCES IN
PROGRAMMING SKILL ACQUISITION

I. INTRODUCTION

What makes a competent computer programmer? What cognitive skills and background knowledge are most important in determining who will succeed and who will not in a computer programming course? There have been numerous psychological investigations of various aspects of computer programming ability (e.g., Brooks, 1977; Jeffries, Turner, Polson, & Atwood, 1981; Mayer, Dyck, & Vilberg, 1986; Pena, 1989). These studies have all looked at a relatively small part of the learning process. Only a few analyses have been conducted across longer periods of learning time (e.g., Anderson, 1990). But thus far, the question of what specific cognitive competencies govern the rate at which programming skill is acquired has not been addressed. The purpose of this study is to examine the development of programming skills over a longer period of learning time.

If you observe a group of individuals learning a new cognitive skill, you will notice some persons learning more efficiently than others. Even under optimal learning conditions using one-on-one tutors, individuals differ widely in how quickly they learn and in how much they remember (e.g., Anderson, 1990; Shute & Glaser, 1990). What causes those differences? Two perspectives have emerged in the individual differences literature. One is the "knowledge perspective" and the other is the "abilities perspective."

Knowledge Perspective

This is the view that cognitive skill results from the possession of vast bodies of domain-relevant knowledge. Chi, Glaser, and Rees (1982) have argued that knowledge is of utmost importance in skill acquisition: "Our hypothesis is that the problem-solving difficulties of novices can be attributed mainly to inadequacies of their knowledge bases and not to limitations in either the architecture of their cognitive systems or processing capabilities" (p. 71). Eight novice-expert studies were reported using physics problems. Novices did not differ from experts on a wide variety of processing measures (e.g., number of categories sorted, identification of key problem features). However, novices did differ from experts in terms of knowledge structures and representations. This was supported by findings of novice-expert differences in inferencing and abstracting capabilities reflecting novices' relative knowledge deficits.

Two experiments were conducted by Walker (1987) investigating whether specific (i.e., baseball) knowledge could compensate for low overall aptitude levels on certain domain-related cognitive tasks (e.g., free recall of a fictitious baseball game). Aptitude levels were estimated from scores on a standard Army aptitude test of general/technical ability. In both experiments, she found that performance on the domain-related cognitive tasks was a function of baseball knowledge, not aptitude level.

A third study investigated the effects of knowledge on cognitive skill acquisition (Schmalhofer, 1982). Computer science and psychology students (neither group having prior LISP familiarity) read a chapter from a programming text on LISP. A posttest revealed that both groups developed an adequate declarative knowledge base of the material presented in the chapter. The psychology students remembered as much of the propositional content as did the computer science students. But, for the psychology students, the knowledge was only declarative. They could not apply the knowledge to problem solving (verifying and debugging LISP programs). In contrast, the majority of computer science students were able to go beyond a declarative understanding and could proceduralize some of the knowledge taught in the text, as indicated by their skill in verifying and debugging LISP programs. This suggests that prior knowledge may determine the degree to which new incoming knowledge about a skill will be proceduralized.

A shortcoming of this perspective that compares extremes of knowledge differences (e.g., novice-expert contrasts) is that in most cases, these studies used subjects that were homogeneous with respect to ability levels (e.g., university students with restricted aptitude ranges). Thus, the studies tended to be biased toward demonstrating that knowledge, not ability, was the reason for the observed individual differences.

Abilities Perspective

An alternative to the knowledge perspective is what might be called the abilities perspective. This is the idea that the most important determinants of skill acquisition are certain cognitive abilities. Ackerman (1988) and Woltz (1988) have investigated the role of general abilities (i.e., working-memory (WM) capacity) on cognitive skill acquisition.[1] Their findings suggest that during the early stages of skill acquisition, WM capacity is the most important determinant of successful learning. Other cognitive ability factors, such as information processing speed, play an important role later in the learning process, after the task has been well practiced. If subjects are prevented from proceduralizing task knowledge, then working memory demands remain high, and WM capacity continues to be a strong determinant of task success, regardless of how much practice subjects have (Ackerman, 1986).

Another study suggests that WM capacity may govern the efficiency of declarative learning (Kyllonen & Christal, in press). They showed that working-memory capacity and reasoning abilities were closely related to one another (correlations in the .80's and .90's). Previous research has established the relationship between reasoning ability and declarative learning (e.g., Snow, Kyllonen, & Marshalek, 1984; Thurstone, 1938). The results from testing various path models using structural equations modeling (EQS) led Kyllonen and Christal to propose that it may actually be WM that is responsible for differences in reasoning ability.

Working-memory capacity has also been shown to be an important predictor of successful learning of a logic gates task (Kyllonen & Stephens, in press). In that study, a working-memory factor predicted all phases of learning success. Furthermore, skill acquisition on the logic gates task showed a changing pattern of relationships between cognitive measures and learning performance variables. Together, "the pattern of changing relationships with cognitive variables over learning phases, and the discontinuities in phase-to-phase performance, reinforce the view that cognitive skill learning has a multidimensional character" (reported in Kyllonen & Woltz, 1989, p. 267). Indeed, it is this "multidimensional character" of skill learning that requires further explanation.

A shortcoming of the "abilities perspective" studies are they mostly involve artificial or novel learning tasks with consequently little or no domain knowledge differences in subjects. So, these studies are more biased toward showing ability, not knowledge, as the reason for ensuing individual differences.

[1] While Woltz does use the term "working-memory capacity," Ackerman more broadly views general ability as (a) the availability of attentional resources and (b) reasoning skills across different content domains. Because attentional resources reflect working-memory capacity, and Kyllonen and Christal (in press) have reported that reasoning skills and working memory are highly correlated, we simplify Ackerman's definition of general cognitive ability by denoting it working-memory capacity.

Other Difficulties with Previous Research

Besides biases, there are two additional problems associated with earlier attempts to ascertain causes of individual differences in skill acquisition. Much of the older, empirical research examined factor-analytically derived constructs such as spatial abilities, reasoning skills, and so on (e.g., Thurstone, 1938) in relation to learning. The first problem is that these factors lacked a process description. This made it difficult to observe or interpret relationships. Thurstone (1947) actually envisioned the factors as stepping stones to more definitive experiments designed to identify and manipulate underlying cognitive processes. But those experiments had to wait several decades until theory and method in psychology caught up with his thinking. Other visionaries also saw the importance of identifying and measuring the processes of learning. For example, Melton (1967) observed, "What is necessary is that we frame our hypotheses about individual differences variables in terms of the process constructs of contemporary theories of learning and performance" (p. 239). Similarly, Cronbach and Snow (1977) stated, "We need more sophisticated hypotheses, based on a careful analysis of the information processing required in the course of learning, and a corresponding analysis of tests to identify the processes that account for high scores" (p. 292).

Today, even though many individual differences researchers examine cognitive processes in relation to learning, a second problem concerns the learning tasks used to study complex skill acquisition. Relatively short laboratory learning tasks are typically employed lasting only a few hours (e.g., Kanfer & Ackerman, 1989; Kyllonen & Stephens, in press; Pellegrino, 1988; Woltz, 1988). An important question is whether these results generalize to more naturalistic learning that occurs over longer periods of time.

Proposed Solution

In this study, we integrate the knowledge and abilities perspectives as determinants of learning because undoubtedly both are important to acquisition. This study teaches a skill that depends on some domain knowledge that people differ on (but does not use extreme cases as with novice-expert contrasts). Furthermore, individuals comprising the sample were heterogeneous in abilities (i.e., the sample came from the general population with a range of aptitudes rather than just university students) so there is no bias toward either of the two perspectives. This study also uses a theoretically-based battery of computerized tests measuring basic cognitive abilities (e.g., WM capacity) in each of three content domains (i.e., verbal, quantitative, and spatial). This stands in contrast to more broadly-defined aptitudes and constructs used in previous individual differences research. Other computerized tests administered in this study measure general and domain-related knowledge. With regard to the problem area involving the use of relatively short-term, artificial learning tasks, this study employs an intelligent tutoring system (ITS) for Pascal programming, delivering up to 30 hours of instruction. The ITS provides an ecologically valid yet controlled means of examining learning in a complex, long-term task. We are not investigating constrained learning as with typical skill acquisition studies (e.g., typing, evaluation of logic gates). Rather, we are looking at complex learning over time involving the interplay between declarative and procedural learning.

Purpose and Approach

The purpose of this study is to ascertain the determinants of individual differences in learning Pascal programming skills: What causes a person to learn faster and better than someone else? We are interested in conducting a more equitable test than has previously been done between knowledge and ability as determinants of skill acquisition. Our approach is to estimate cognitive abilities, incoming knowledge, and declarative and procedural learning processes, then link cognitive abilities and knowledge to the learning processes. The sets of estimates are related using exploratory and confirmatory statistical modeling techniques including structural equations modeling.

Research Questions

The questions driving this study can be summarized as follows: (a) What are the relative contributions of incoming knowledge and cognitive abilities in determining different learning factors? (b) Are declarative and procedural learning factors distinct or best defined as a single, general learning factor? and (c) Is transfer predictable? If so, what underlies it? Other research questions focus on what individuals actually do in the tutor that may influence learning.

II. METHOD

Subjects

Subjects in this study consisted of 260 students participating in a 7-day (40-hour) study on acquisition of Pascal programming skills from an intelligent tutoring system. There were 50 females and 210 males in the sample. Subjects were recruited and selected from San Antonio colleges, technical schools and universities to match demographic and general ability characteristics of the Air Force enlisted population. All subjects were high school graduates (or equivalent) with a mean age of 22. None of the subjects had any prior Pascal programming experience and all subjects were paid for their participation.

Apparatus

Experimental cognitive abilities tasks were administered on Zenith 248 microcomputers with standard keyboards and EGA color video monitors (640 x 350 resolution). Software was written to achieve millisecond timing for response latency measures. The complex learning task was administered in an adjacent facility on Xerox 1186 computers with standard keyboards and high resolution (1024 x 840) monochromatic displays on 19" monitors. Software was written in Interlisp-D and LOOPS.

Procedure

Subjects were tested in groups of 15 to 20 at Lackland Air Force Base, Texas, in two testing facilities: the Cognitive Abilities Measurement (CAM) laboratory, and the Complex Learning Assessment (CLASS) laboratory. In each facility, subjects occupied individual testing carrels and instructions, testing, and feedback were computer administered with proctors available to answer questions. On the morning of Day 1, subjects were given a brief orientation to the entire study. Subjects were then administered 6 hours of cognitive ability tasks, with short breaks between tasks and a 1-hour break at noon. On subsequent days, subjects were provided with instruction and practice in Pascal programming through an ITS. On the morning of the 7th day (excluding weekends) of each subject's participation, another battery of cognitive ability tasks was administered.

Cognitive Tasks (CAM laboratory)

To measure cognitive abilities, the CAM-1 battery was administered (Kyllonen & Christal, 1990). This battery consists of computerized tests measuring working memory (WM) capacity, processing speed (PS), general knowledge (GK), and procedural knowledge (PK). Tests for WM, PS, and PK were developed in each of three domains: quantitative, verbal, and spatial, while GK tests were measured in the verbal domain. In the past, researchers have demonstrated that individual differences in a wide variety of learning tasks can be accounted for by these cognitive factors (see Kyllonen & Woltz, 1989 for an overview). Examples from CAM-1 tests follow while details of all tests[2] can be found in the Appendix.

Working Memory (WM)

Subjects had to learn and remember numeric values assigned to the letters A, B, and C. Three statements (e.g., A = B/3, B = C + 2, C = 4) were presented one at a time, and subjects could look at each one for as long as they wanted before going on to the next statement. Subjects were then asked to recall the values of the letters, one at a time (e.g., B = ? C = ? A = ?). Previous analyses of this type of test showed that errors are a function of the concurrent storage and processing requirements.

Processing Speed (PS)

Two words were simultaneously presented on the computer screen and subjects had to decide whether the words had similar or different meanings (e.g., humid--moist). Subjects typed in either "L" (like) or "D" (different). The level of stimuli difficulty was purposefully low (i.e., high word familiarity) so as not to confound the issues of processing speed and existence of knowledge. Latencies were recorded in milliseconds.

General Knowledge (GK)

Subjects were asked general questions (e.g., What is the process by which plants make food from sunlight?) and had to respond by typing in the first two letters of the answer (e.g., "PH" or "FO" for photosynthesis). Accuracy and latency were used as measures of general knowledge.

In addition to the CAM-1 battery, we also administered a battery of domain-relevant knowledge tests that we created just for this study. This battery measuring *specific knowledge (SK)* consisted of three tests. The first two tests measured incoming knowledge about simple math and programming constructs (e.g., Which of the following items are examples of an INTEGER? (a) -1.25; (b) -67; (c) four; (d) 999.9; (e) 100000; (f) 1; (g) 34.2; (h) 369). The third test measured comprehension of errors in Pascal code (see Appendix).

Complex Learning Tasks (CLASS Laboratory)

Declarative learning processes were obtained from performance on the Pretutor (i.e., computer-aided instruction teaching simple math and programming concepts). Procedural learning processes were estimated from performance on the Pascal tutor (e.g., time to complete programming problems). Transfer, a third learning process, was measured from performance

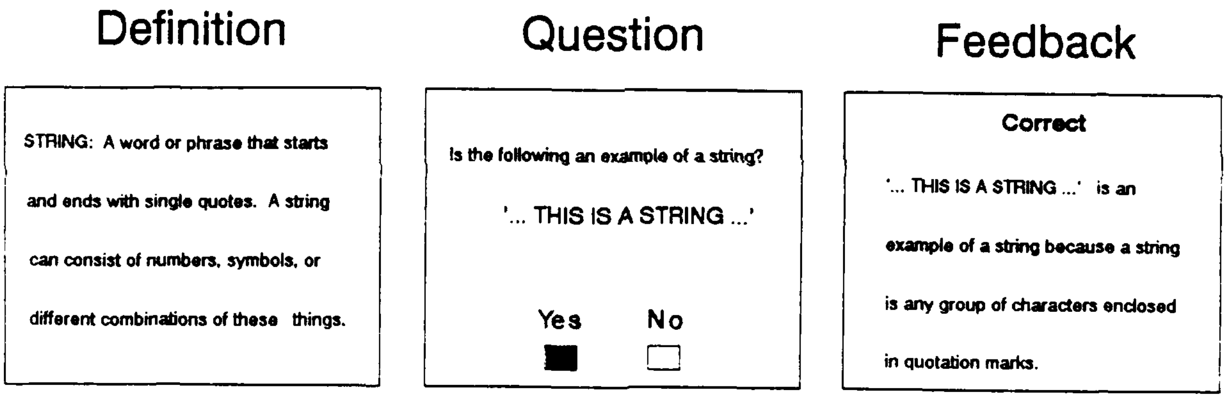
[2] We do not report examples or data from the procedural knowledge (PK) tests because they were so highly correlated with WM tests.

on a criterion test battery administered at the conclusion of the tutor. Each of these learning processes will be discussed in turn.

Declarative Learning (Pretutor)

Subsequent to the administration of the cognitive abilities tasks, but prior to learning from the Pascal ITS, subjects were administered computer-based instruction covering simple programming and math concepts, namely: integer, real number, string, data, sum, product, constant, variable, expression and value assignment. This instruction was developed in response to having tested about 200 pilot subjects on the Pascal tutor and discovering that some subjects had difficulty learning the programming curriculum because they lacked knowledge presumed by the system (e.g., not knowing what an integer was).

Subjects received initial definitions of concepts (which they could view as long as needed), followed by eight questions (a "block") pertaining to the concept. After each question, and regardless of accuracy, feedback was provided (see Figure 1). A strict learning criterion was imposed (i.e., 100% correct on two successive blocks of 16 items for a given concept), and concepts dropped out of the study-test cycle only when the criterion was met.



Definition

STRING: A word or phrase that starts and ends with single quotes. A string can consist of numbers, symbols, or different combinations of these things.

Question

Is the following an example of a string?

'... THIS IS A STRING ...'

Yes ■ No □

Feedback

Correct

'... THIS IS A STRING ...' is an example of a string because a string is any group of characters enclosed in quotation marks.

Figure 1. **Pretutor: Example Definition, Question, and Feedback.**

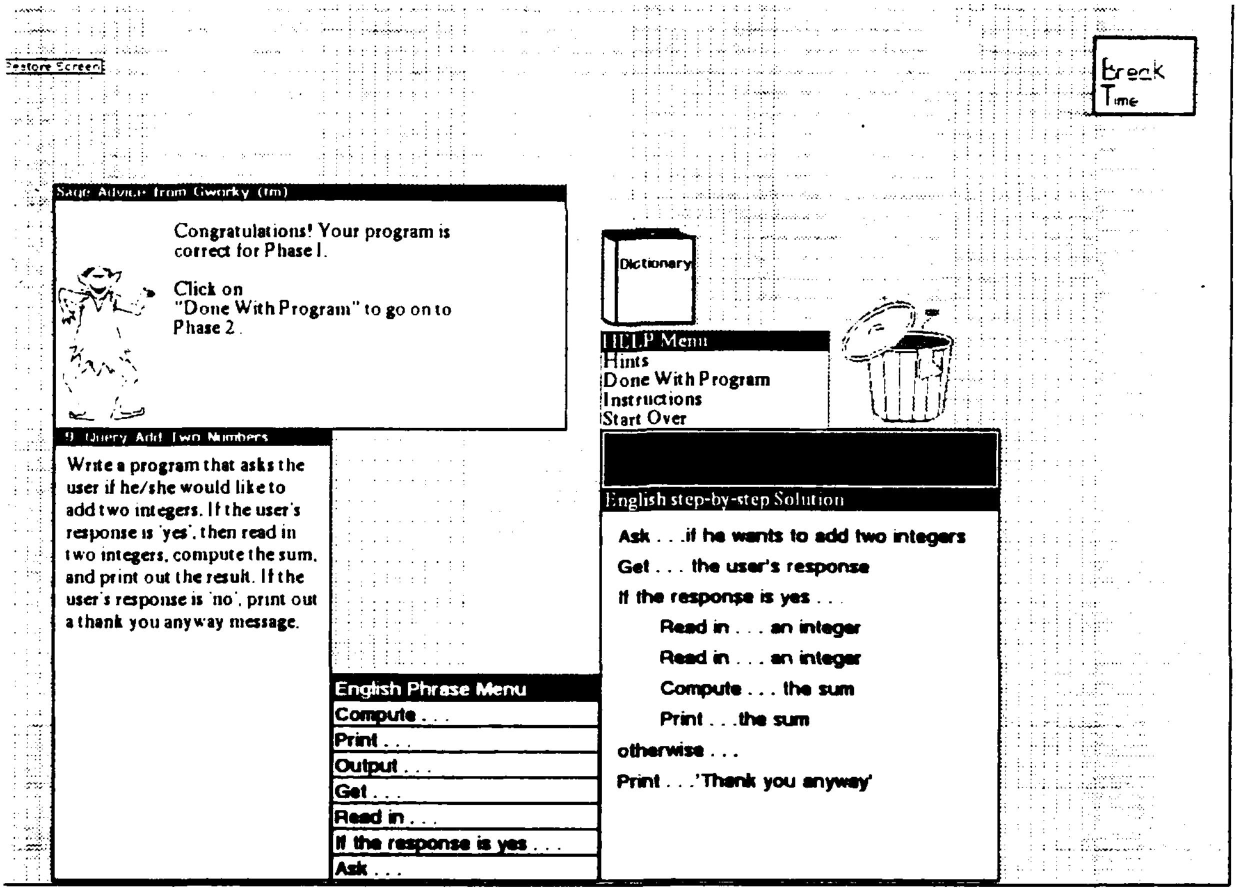
Declarative learning indicators were: (a) Time to complete the Pretutor and (b) Overall accuracy on the Pretutor. Both reflect the speed and accuracy of acquiring new concepts.

Procedural Learning (Pascal ITS)

Subjects spent up to 30 hours learning from "Bridge," the Pascal ITS. This system was originally developed by Bonar and his staff (Bonar, Cunningham, Beatty, & Weil, 1988). We modified the tutor to generate computer-tallied learning indicators summarizing actions from the student history list (cf. Kyllonen & Shute, 1989). Other additions included: an on-line dictionary containing relevant programming terms, use of single quotes surrounding strings, three hints per problem increasing in explicitness, elimination of infinite loops from certain problems, simplification of the language used in the hints and feedback, elimination of inconsistencies between phases, delineation of different errors types, and so on.

The tutor's design originated from research conducted with "programming plans" (e.g., Soloway, Bonar, & Ehrlich, 1983; Spohrer, Soloway, & Pope, 1985). Programming plans are high-level concepts and techniques relevant to introductory programming. Examples include: input plan, counter plan, and constant running total plan. Plans putatively allow a beginning programmer to represent goals of a programming task. The tutor provides a means of translating those goals into code.

The curriculum embodied by the ITS corresponded to about one half a semester of introductory Pascal programming (J.G. Bonar, personal communication, March 1990) and consisted of 25 programming problems. These problems became increasingly more complex, from simple problems (e.g., *Write a program which prints out your name*) to problems involving complex "While" and "Repeat until" loops. Each problem required a three phase solution from informal to formal specification. Phase 1 required a natural language solution to a given problem, Phase 2 involved generating a visual solution by sequencing programming plans like a flow chart or jigsaw puzzle, and finally, Phase 3 involved translating the visual solution into Pascal code. Subjects were required to successfully complete each phase in sequence within a problem before moving on to the next problem. The three phases are illustrated in Figures 2, 3, and 4.



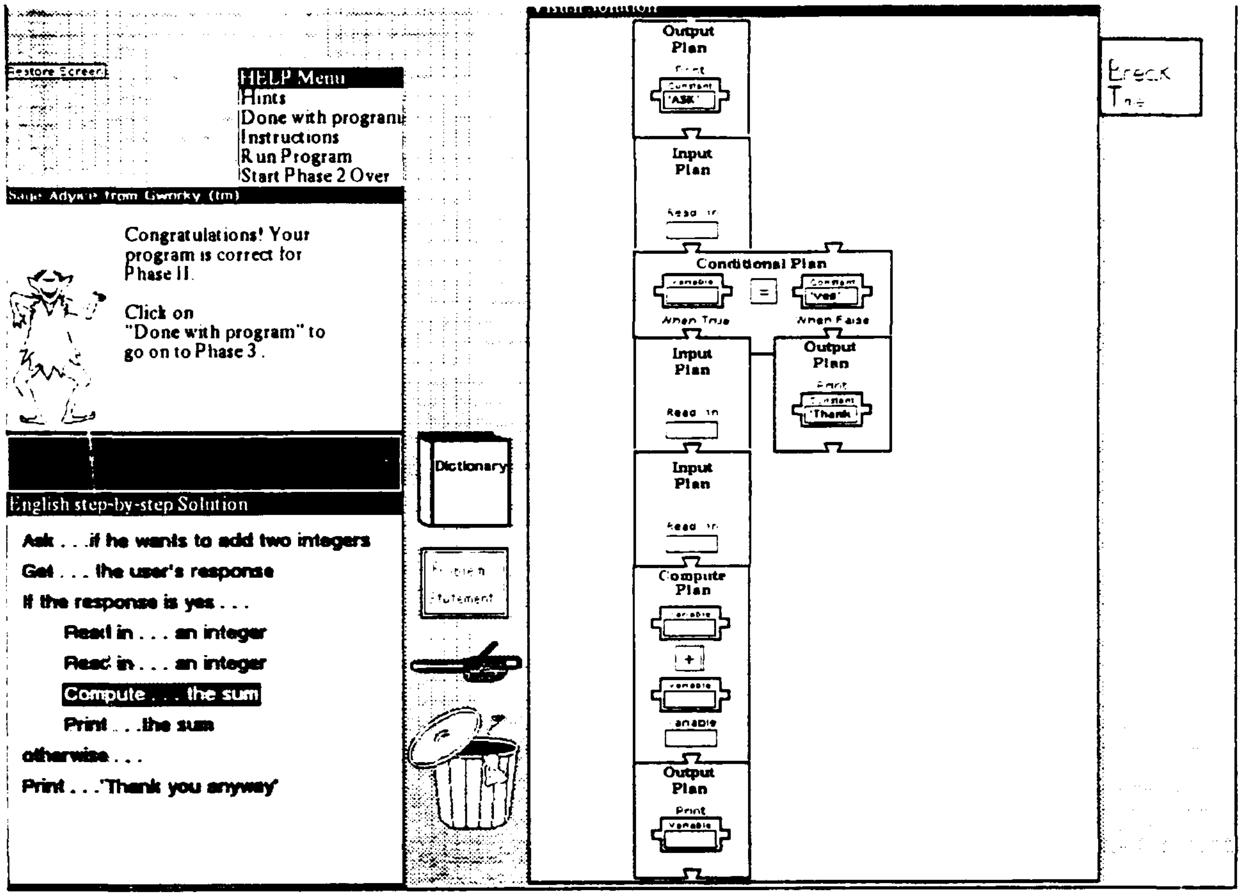
Congratulations! Your program is correct for Phase I.

Click on "Done With Program" to go on to Phase 2.

Write a program that asks the user if he/she would like to add two integers. If the user's response is 'yes', then read in two integers, compute the sum, and print out the result. If the user's response is 'no', print out a thank you anyway message.

```
Ask . . .if he wants to add two integers
Get . . . the user's response
If the response is yes . . .
    Read in . . . an integer
    Read in . . . an integer
    Compute . . . the sum
    Print . . .the sum
otherwise . . .
Print . . .'Thank you anyway'
```

Figure 2. **Phase I Example of the Pascal ITS.**



Congratulations! Your program is correct for Phase II.

Click on "Done with program" to go on to Phase 3.

English step-by-step Solution

Ask . . .if he wants to add two integers
Get . . . the user's response
If the response is yes . . .
    Read in . . . an integer
    Read in . . . an integer
    Compute . . . the sum
    Print . . .the sum
otherwise . . .
Print . . .'Thank you anyway'

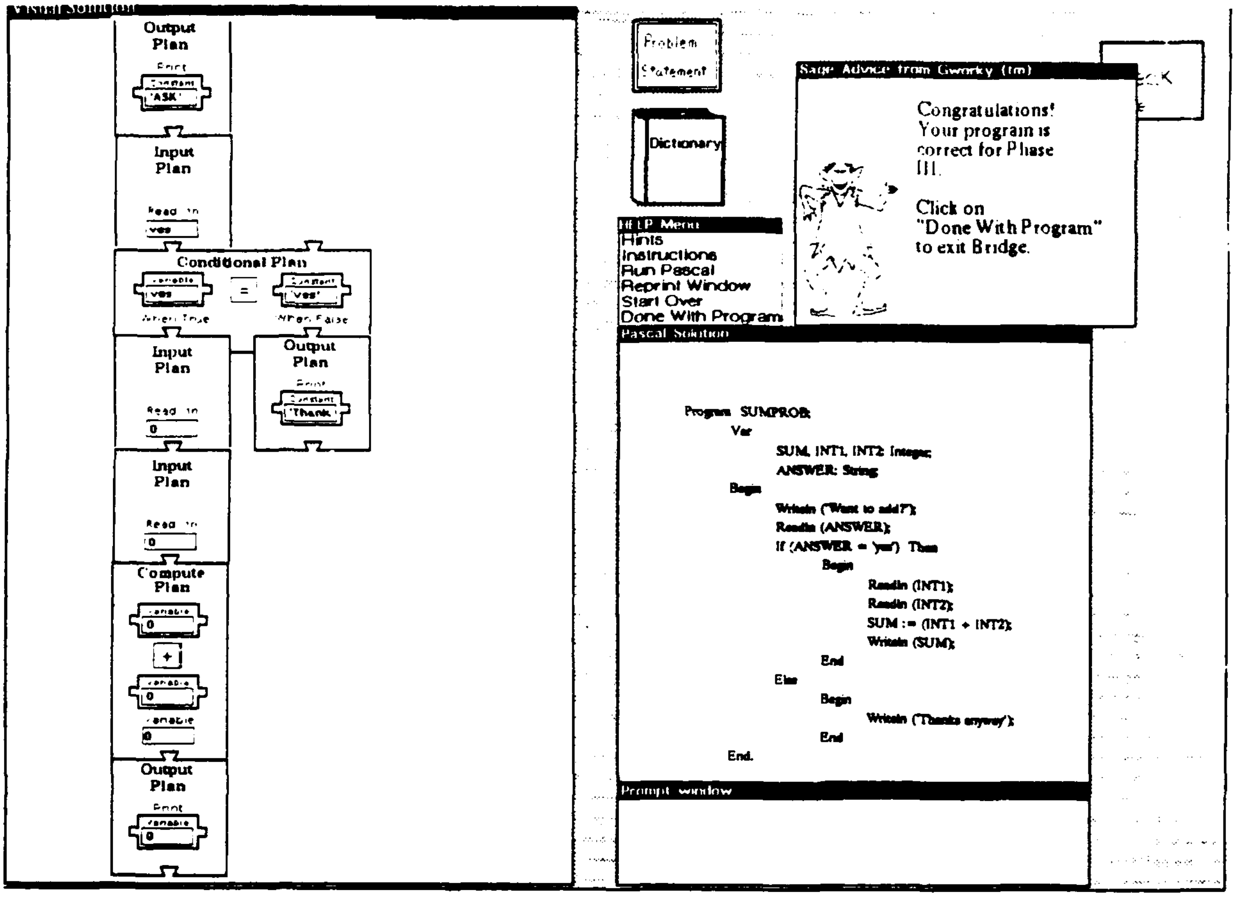
Figure 3. **Phase II Example of the Pascal ITS.**

Context sensitive hints were always available for subjects to use or abuse.[3] They ranged from initially vague to unambiguous. Another feature of the ITS was that programs created in Phases 2 and 3 could be run, and the execution of the program could be observed. A large set of learning indicators were extracted from the student history, a collection of all actions taken during ITS involvement. Some example indicators, per problem, per phase, are seen in Table 1. The choice of learning indicators used in this study resulted, in part, from past research employing indicators that were shown to capture a large amount of learning variance (e.g., Shute & Glaser, 1990).

These data provide a wealth of information regarding a student's understandings and misunderstandings at any given time. But to prevent drowning by indicators, we selected three measures as the most informative and also that mapped onto the indicators measuring declarative learning: (a) Time to complete the tutor where, similar to the Pretutor, this measure represented learning efficiency as it involved both speed of acquisition and accuracy on the problems; (b) Slope of hints requested across 25 problems, implying the presence or absence of procedures

[3] Some individuals who relied on hints rather than figuring out solutions themselves significantly reduced their final transfer measures.

over time; and (c) Slope of conceptual errors made across 25 problems, denoting changes in the soundness of procedures. The slope measures for hints and errors were better (i.e., more predictive) parameters of learning than either total counts or starting values (i.e., intercepts). The most effective estimate of errors was a subject's factor score on a general error factor (Note: a factor analysis was computed on the five error types and resulted in a single factor solution). Slope and intercept measures were derived from individual factor scores.



Congratulations! Your program is correct for Phase III.

Click on "Done With Program" to exit Bridge.

```
Program SUMPROB;
    Var
        SUM, INT1, INT2: Integer;
        ANSWER: String
    Begin
        Writeln ('Want to add?');
        Readln (ANSWER);
        If (ANSWER = 'yes') Then
            Begin
                Readln (INT1);
                Readln (INT2);
                SUM := (INT1 + INT2);
                Writeln (SUM);
            End
        Else
            Begin
                Writeln ('Thanks anyway');
            End
    End.
```

Figure 4. **Phase III Example of the Pascal ITS.**

Table 1. **Examples of Performance Indicators**

Number of times hints were requested
Number of times hints were received (unsolicited)
Number of times a problem was started over
Number of times the solution was run (total runs, successful runs, crashes)
Number of times the problem statement was viewed
Number of errors of omission (leaving out a phrase, plan, or line or code)
Number of errors of sequencing (misplaced phrase. plan, or line of code)
Number of errors of logic (e.g., incorrect stopping condition of a loop)
Number of operator errors (incorrect use of an operator)
Number of data type errors (e.g., input a real number instead of an integer)

Transfer of Learning (Criterion Posttest Battery)

Transfer, in this study, was defined as the ability to apply knowledge and skills acquired from one learning task (i.e., the tutor) to new situations. The associated learning processes include: retention, application, and generalization. Subsequent to completing all 25 problems in the ITS curriculum, subjects completed an on-line battery of three tests measuring the breadth and depth of knowledge and skills acquired from the tutor. Each test consisted of 12 items isomorphic to tutor problems, reflecting the range of problem difficulty encountered in the tutor.

Retention Test. The first test (retention) was created to be a recognition test requiring error detection for existing Pascal code. Subjects saw a problem statement, then Pascal code, and a menu of possible error types (i.e., 1 line missing; 2 lines missing; error in line; misplaced line; unnecessary line). They had to indicate the type of error in the code (if any) and then select the line of code that was at fault. For each item, the score reflected both whether they identified the correct error type and whether they identified exactly where in the code the error occurred.

Application Test. The second test (application) involved rearrangement of code -- decomposing problem statements into general Pascal commands and sequencing them into a solution for the programming problem. Here the subject received a problem statement (e.g., "Write a program that multiplies the integers from one to some number that is specified by the user. Then have the program print out the result."). From a menu of different Pascal statements, the subject selected the relevant ones and arranged them in the solution window. The score reflected how many errors subjects committed in their rearranged solution (e.g., missing begin/end loop, wrong command, wrong command argument, failure to initialize a variable, line missing, misplaced line, unnecessary line).

Generalization Test. Finally, the third test (generalization) involved writing Pascal solutions to programming problems from scratch, going beyond what was required by the menu-driven tutor. Subjects were shown a problem statement (e.g., "Write a program that computes the sum of integers from 10 to 50. After calculating the result the program should print it out."). Then they were required to type the solution into an on-line text editor, without any tutor support. The scoring procedure was similar to that from the Application test, except that there were additional types of errors that were possible (e.g., incorrect loop, missing quotes, missing identifier, using unnecessary variable, wrong use of an operator, forgetting to declare a variable, missing quotes on a string). A rational weighting scheme reflecting error severity was employed.

III. RESULTS

Means and standard deviations of the major variables are presented in Table 2. Accuracy (percent correct) on the Pretutor was fairly high (M = 84.4%; SD = 8.7). However, the overall accuracy on the final transfer tasks (i.e., average of the three tests) showed large individual differences. The range in scores was 17.3% to 96.7%, with a mean of 56 and a standard deviation of 19. These data were normally distributed and showed almost a 6:1 ratio in performance differences. The time to complete the Pretutor ranged from 23 minutes to 2.8 hours (7:1 ratio between fastest to slowest learners). Even more extreme, time to complete the tutor ranged from 2.8 hours to 29.2 hours (10:1 ratio).

Throughout the ITS curriculum, programming problems became progressively more difficult while subjects became progressively more facile with the Pascal constructs and system interface. Because individuals could not proceed to the next problem until completely successful in the current problem, their total time to go through the ITS curriculum was a rate measure of learning, involving both speed and accuracy. When the time data were clustered by problem types, they showed a general time reduction for subsequent, similar problems. Figure 5 shows the first 19 Pascal problems of the tutor, separated by problem type. Only the first 19 of 25 total problems were used in this graph because the remaining problems involved combinations of earlier problem types.

Table 2. **Summary Statistics (N = 260)**

Variable	Mean	SD	Min	Max
Pretutor time (minutes)	100.5	36.3	22.6	168.1
Pretutor accuracy (% correct)	84.4	8.7	48.4	96.8
ITS time (hours)	12.2	5.2	2.8	29.2
Hints total (across 3 phases)	755.8	613.1	4.0	3131.0
Hints slope (across 3 phases)	30.1	24.7	0.2	133.9
Errors total (across 5 error types)	991.1	741.6	20.0	3791.0
Errors slope (across 5 error types)	39.7	29.7	0.8	156.0
Transfer (% correct)				
retention	38.4	23.3	0.0	100.0
application	76.1	15.7	12.0	99.0
generalization	52.3	24.5	0.0	99.0
average	55.8	19.0	17.3	96.7

Note. For the descriptive data on errors, we report total number of errors across the five error types and the total slope of the errors. Factor analysis (principal axis factoring with varimax rotation) on the five errors produced one general error factor. Factor loadings on this general error factor were used in the structural equations modeling with standard values (i.e., mean = 0, SD = 1).

We tested path models with latent factors using structural equations modeling. Alternative models were contrasted that reflected different psychological assumptions. Maximum likelihood estimates of model parameters and goodness-of-fit indices were obtained between the sample variance/covariance matrix for all variables and the best-fitting recovered matrix, using the EQS 3.0 computer program (Bentler, 1989). EQS implements the Bentler and Weeks (1980) approach to the analysis of linear structural equation systems and is similar to the LISREL computer program (Joreskog & Sorbom, 1988). This program allows for multiple regression, path analysis, simultaneous equations, first-and higher-order confirmatory factor analysis, as well as regression and structural relations among latent variables.

In the first model tested, Model 1, all of the cognitive ability factors have been linked with each learning factor. The research questions to be answered include: (a) What are the relative contributions of incoming knowledge and cognitive abilities in determining learning factors? (b) Are declarative and procedural learning factors distinct or best defined as a single, general learning factor? and (c) Is transfer predictable? If so, what underlies it?

As seen in Figure 6, under the declarative and procedural learning factors, arrows point from the factors to their indicators. A hierarchical structure was created involving the declarative learning factor with arrows emanating to all five indicators. This may be viewed as a general associative learning factor, assumed to underlie all indicators. That is, performance on any indicator was assumed to depend partly on how much declarative knowledge has been acquired about the skill being learned. A second factor, procedural learning, was posited, involving an individual's ability to apply knowledge in creating more efficacious procedures. The procedural learning factor was assumed to underlie only the indicators of problem solving in the ITS and to be orthogonal to the declarative learning factor.

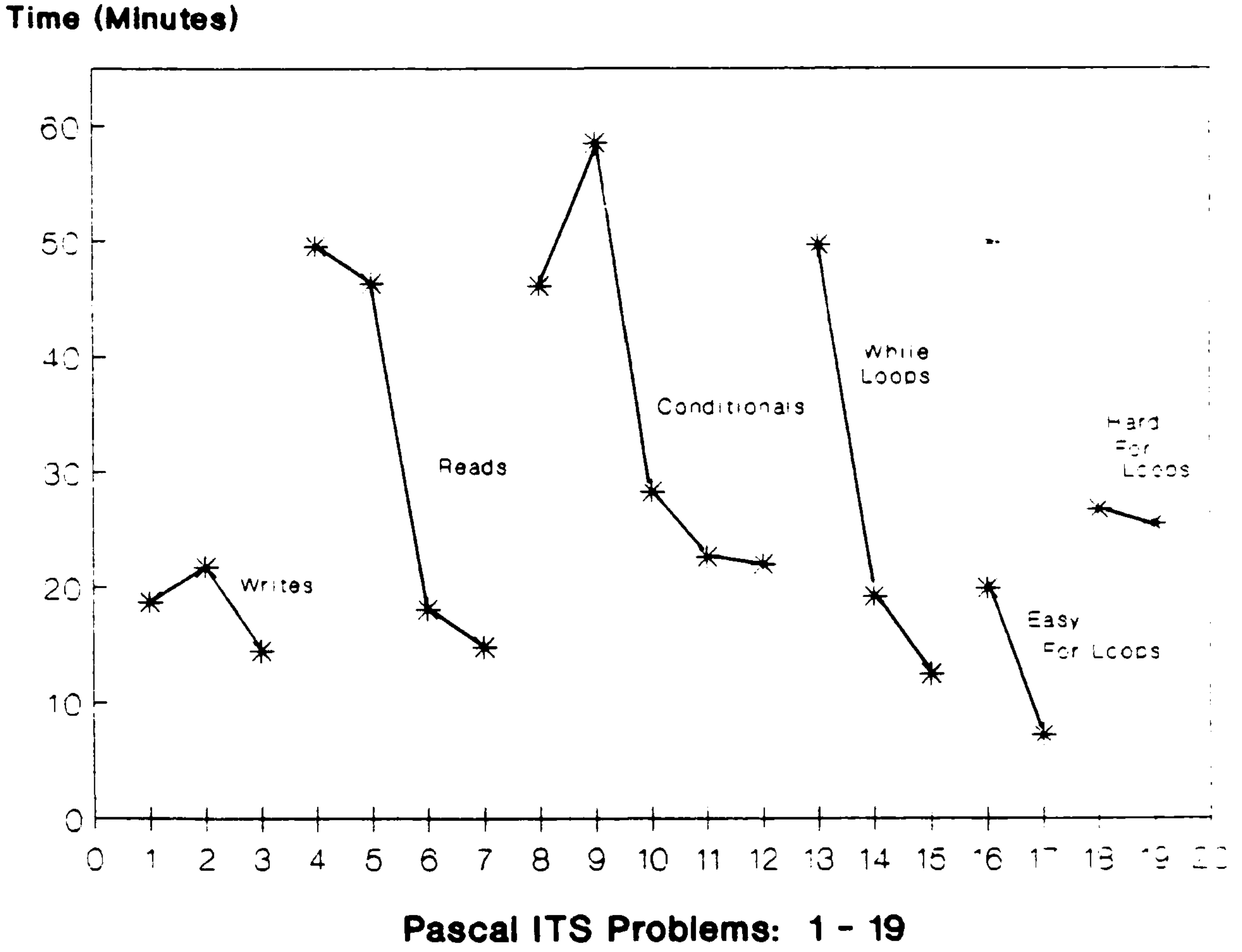


Figure 5. **Time on Pascal Problems Separated by Problem Type.**

Knowledge and Ability Correlates of Programming Skill Acquisition

Figure 6 shows the results of data applied to Model 1. Solid arrows convey significant relationships while dashed arrows depict nonsignificant relations. The numbers on the arrows are regression coefficients.[4] The Bentler-Bonett Nonnormed Fit Index (NNFI) = .964, N = 260, thus the model fits the data rather well.

[4] These numbers denote both the strength and directionality of relationships as well as the unique influence of one element on another (partial correlations).

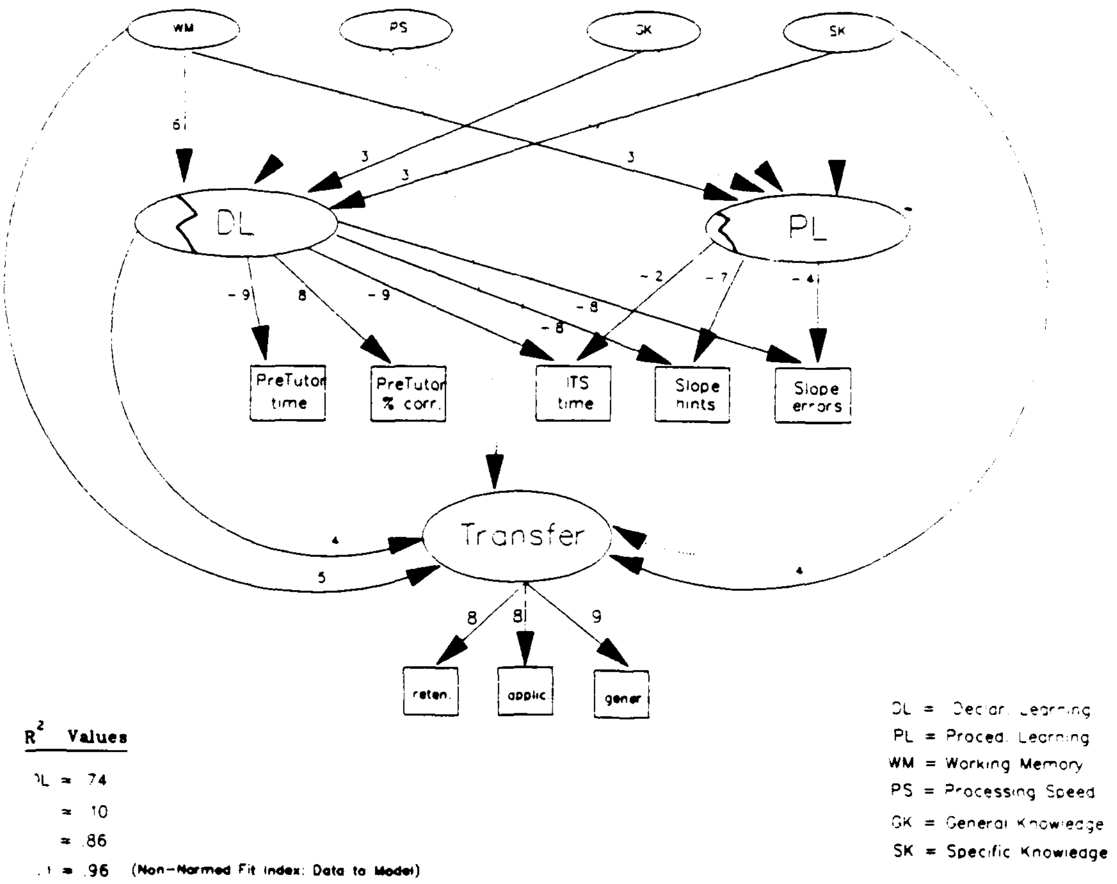


Figure 6. **EQS Solution on Model of Learning.**

There appear to be two answers for the question addressing the relative contributions of incoming knowledge versus ability in determining learning. First, with regard to *declarative* learning, both knowledge and cognitive abilities are important. Working-memory (WM) capacity, general knowledge (GK), and specific knowledge (SK) together accounted for a considerable amount of variance in the latent declarative learning factor ($\underline{R}^2$ = .74).[5] This analysis also indicated that: (a) information processing speed was not an important predictor of declarative learning; and (b) of the significant predictor variables, WM played the more important role in predicting declarative learning evidencing twice the strength of both general and specific knowledge measures. Second, in contrast to declarative learning, only 10% of the variance in the latent procedural learning factor was accounted for. Only WM was significantly related to this factor.

[5] Note that we are not claiming that we accounted for 74% of the variance in the observed measures (i.e., the criteria usually associated with the reporting of R^2 values), only 74% of the variance in the latent factor (a criterion less often discussed in reference to R^2 values). Factor variance represents the elimination of two sources of variance in indicators: variance due to test unreliability (measurement error) and variance due to test uniqueness. Thus, unless indicators are perfectly reliable (no error) and perfectly intercorrelated (no uniqueness) R^2 for a factor will be higher than R^2 for an indicator of that factor.

The next question focused on the uniqueness of the declarative and procedural factors. With EQS, it is possible to test alternative models and compare goodness-of-fit indices. Model 2 was defined exactly the same as Model 1 only with procedural learning removed as a factor. EQS was computed for Model 2, and the data compared. For Model 1, $X^2_{(101)} = 182.2$, and for Model 2, $X^2_{(107)} = 263.9$. The difference between these two models was $X^2_{(6)} = 81.7$, $p < .001$. Therefore Model 1, specifying procedural learning as a separate, additional factor, turned out to be more appropriate -- significantly fitting the model better for these data.

Another observation supported the hypothesis that declarative and procedural learning are distinct. Different patterns of predictors defined the two learning factors. Only WM predicted procedural learning whereas WM, GK, and SK predicted declarative learning. This indicates two different factors, each being predicted by a different arrangement of cognitive measures. Finally, the factor loadings on the procedural learning factor were all significant. This lends additional support for the distinctiveness of the procedural learning factor, apart from a general, declarative (or associative) learning factor.

The last question was whether transfer was predictable. The answer is an unequivocal "*yes*." Three factors, general working-memory (WM) capacity, incoming domain-specific knowledge (SK), and domain-specific learning proficiency accounted for almost all ($R^2 = .86$) of the variance in the latent retention/transfer factor.[6] The SK and WM factors alone accounted for 70% of the variance in the transfer factor. Thus, a large part of programming skill acquisition is due to knowledge and ability factors brought to the learning task! Furthermore, the efficiency by which an individual acquires domain-specific declarative knowledge is a strong predictor of transfer while procedural learning efficiency did not predict transfer in these analyses.

Learning Behavior Correlates of Programming *Skill Acquisition*

Another way of looking at the research question of "who did well and why" involves characterizing successful versus less successful learners by their learning behaviors. In other words, what did the more successful individuals do during the learning process? To address these issues we conducted a series of exploratory analyses of the relationships between indicators of learning behavior and performance scores from the tutor and transfer tasks. (Because these were exploratory analyses, we report simple and multiple correlations with observed rather than latent variables throughout this section.)

As indicated earlier, learning indicators were tallied across 25 problems, in each of three phases per problem. Often, the value of a collapsed learning indicator turned out to be misleading or inadequate in isolation, failing to capture potential changes across time. For example, one gross level indicator, "*busy*," was defined as the number of actions made (i.e., keystrokes or menu item selections) per minute. The correlation of this measure with an overall transfer index (i.e., mean of the three criterion tests) was $r = -.31$ ($p < .001$). Being generally busy, it seemed, was not conducive to success on the criterion test battery. Similarly, the correlation of busy to total time on the tutor was .08 (ns) leading to the inference that activity level is simply not linearly related to learning efficiency in this environment. However, when a slope value for busy across the 25 problems was derived, the correlations between the slope

[6] In Figure 6, the arrow from declarative learning to transfer comes from the part of the factor (to the left of the 'crack') representing the proportion of variance not accounted for by cognitive abilities or knowledge. Because 74% of the variance of declarative learning was accounted for, 26% remains unexplained. So, it was this residual variance of declarative and procedural learning used to predict transfer. Note that the residual variance in procedural learning did not significantly predict transfer.

of "busy" and both transfer and time-on-tutor were -.64 and .54 ($\underline{p}$ < .001), respectively. Furthermore, when the two parameters of busy (i.e., slope and mean) were entered into a regression equation predicting transfer, both remained in the equation ($\underline{p}$ < .001) after backwards elimination ($\underline{R}^2$ = .44).

One aspect of successful learning is that it is better to be busy early, engaging in orienting behaviors (e.g., exploring, testing) but becoming more focused over time. This pattern is clearly illustrated in the following correlations. Data from the 25 programming problems were divided into fifths yielding five blocks, five problems per block. A "busy" value was computed for each block. In Table 3, correlations are shown between blocks (first to last) with overall transfer and time-on-tutor.

Table 3. **Correlations Between "Busy" with Transfer and Time on Tutor**

Variable	**Transfer**	**Time**
Busy1	.33**	-.48**
Busy2	-.02	-.17*
Busy3	-.30**	.11
Busy4	-.40**	.24**
Busy5	-.50**	.33**

Note. N= 260; *p < .01; **p < .001.

What is striking about this pattern is the reversal of correlations for each dependent measure. Initially, being busy is a positive behavior, associated with significantly higher transfer scores and less time on the tutor. But later, significant negative relationships are seen between busy and these same dependent variables.

Requesting hints from the system (total number of hints) and asking for hints over time (slope of hints) showed the same kind of differential patterns of relations with a general transfer score. When a collapsed transfer accuracy score was regressed on these two variables, both remained in the equation ($\underline{p}$ < .01) after backwards elimination ($\underline{R}^2$ = .46). Here, fewer total hints requested were associated with better performance on the criterion test battery ($\underline{r}$ = -.64; $\underline{p}$ < .001 between total hints and overall transfer). Moreover, if a subject needed to request hints from the system, it seems to have been more fruitful to have asked for them early on in the tutor ($\underline{r}$ = -.67; $\underline{p}$ < .001 between slope-of-hints and overall transfer). Again, a negative slope measure predicted successful Pascal learning.

The third learning behavior showing differences across time concerns running programs (i.e., in Phases 2 and 3 of the tutor). The simple correlations between total number of runs and the slope of runs were .27 and .31, respectively ($\underline{p}$ < .001). The slope measure correlated *positively* with transfer task performance, suggesting that running a program was a relatively productive behavior, especially for the more complex problems in the latter stages of the tutor (i.e., the better students would run their problem solutions, especially the more complex problems). As with the hints and busy variables, when transfer was regressed on runs and the slope of runs, both remained in the equation ($\underline{p}$ < .001) after backwards elimination ($\underline{R}^2$ = .16).

Knowledge, Cognitive Abilities, Learning Behaviors, and Transfer

Ability factors predict transfer and learning behaviors predict transfer. What is the relationship between ability factors and learning behaviors? Perhaps learning behaviors predict transfer by virtue of their correlation with ability factors. That is, people may take hints (which negatively predicts transfer) because of insufficient WM capacity (which also predicts transfer). Simple correlations are shown between abilities (WM and PS), incoming knowledge (GK and SK) and activity level (Busy 1 to 5) in Table 4. Do any of the ability factors show pattern(s) of relations with "busy" over time?

Table 4. **Correlations of Abilities and Knowledge by Activity Level Across Time**

Variable	WM	PS	GK	SK
Busy1	.14	-.31**	.06	.24**
Busy2	-.09	-.05	-.19*	-.08
Busy3	-.29**	-.08	-.33**	-.27**
Busy4	-.33**	.02	-.36**	-.32**
Busy5	-.45**	.11	-.44**	-.44**

Note. N= 260; *p < .01; **p < .001.

Consider the pattern of correlations over time with the specific knowledge (SK) factor. At the outset of learning (Busy1), learners with more specific knowledge were significantly more active in exploring the environment. Across time they engaged in progressively fewer exploratory activities, perhaps becoming more focused and careful. As Table 4 shows, the other cognitive factors showed similar reversals.

Is *there any correlation* between learning behaviors and transfer once the variance associated with knowledge and abilities is removed? The *unique* relationship between busy, hints and runs to overall transfer task performance is illustrated in Table 5, below. Transfer, in this analysis, was the average score from the three post-tutor outcome tests (retention, application, and generalization). Semi-partial (i.e., part) correlations were computed to represent the relationship between each learning behavior and transfer when individual correlations to WM, PS, GK, and SK were removed from the predictor variables.

Table 5. **Semi-partial Correlations of Learning Behaviors with Overall Transfer, Controlling for Ability**

Time	Busy	Hints	Runs
Time 1	.15*	-.18*	.02
Time 2	.04	-.17*	.15*
Time 3	-.08	-.21**	.15*
Time 4	-.15*	-.27**	.15*
Time 5	-.16*	-.28**	.28**

Note. N= 260; *p < .01**; p < .001.

Several observations are worth noting from this table. First, even when all four of the cognitive ability factors were partialled out of the predictor variables, there still remained significant relationships between the learning behaviors and transfer. Second, the activity level behavior (busy) still showed a reversal pattern whereby initial levels of activity were positive, but later, more actions were negatively correlated with transfer performance. This represents a more "pure" learning style measure after the knowledge and cognitive ability variance was removed. Hints and runs showed opposite correlation patterns with one another. Asking for hints was progressively more harmful and running programs was progressively more productive in relation to programming skill acquisition.

Analyses of relations among incoming knowledge, cognitive ability factors, learning behaviors, and transfer suggest that Pascal programming skill acquisition is mediated by incoming knowledge and abilities differentially at different times during the learning process. Specific learning behaviors, like asking for hints or running programs, appear to mediate cognitive skill acquisition, being more influential at certain times during the learning process than at other times. Moreover, learning behaviors appear to influence transfer over and above that of the cognitive ability measures.

IV. DISCUSSION

The main research question explored in this paper concerned the nature and magnitude of relationships among cognitive ability measures, incoming knowledge, and the learning processes underlying programming skill acquisition. First, consider the debate involving incoming knowledge versus cognitive abilities as determinants of learning a new cognitive skill. For this task, the answer was that both factors were almost equally important. Furthermore, the pattern of relationships indicated that for declarative learning, WM, GK, as well as SK were significant predictors of efficient knowledge acquisition while only WM seemed to impact procedural learning. An explanation of these findings would be that the more knowledge one has at the outset of learning, the richer the network of concepts and associations into which new declarative knowledge may be incorporated, subject to mediation by working-memory capacity. In accord with the findings of Ackerman (1988) and Woltz (1988), there was a significant effect of working-memory capacity on declarative learning -- the early stage of cognitive skill acquisition. Unlike their findings, however, in the latter stages of skill acquisition, subjects in the current study never became functionally automatic, nor did PS ever enter as a significant predictor of the procedural learning factor.[7]

Why was there such a low accounting of procedural learning ($\underline{R}^2 = .10$)? One explanation is that the tutor's curriculum was structured so that subjects were constantly learning new concepts and procedures. Thus they never quite reached expertise for any given procedure before being confronted with something new to learn (e.g., the construction of a "While loop"). About every third or fourth problem (see Figure 5), the tutor introduced a new programming construct or procedure. Past research (e.g., Woltz, 1988) has shown that the unique aspects of procedural learning do not usually emerge until WM decreases and PS increases in predictive importance. This changing pattern of relations only happens following sufficient time and consistent practice with a new skill. The reason for the low prediction of procedural learning, as well as the fact that PS did not enter into the predictive equation was most likely due to the constant introduction of new procedures by the tutor. And, as mentioned earlier, if subjects are not allowed to fully proceduralize task knowledge, then WM demands remain high and it is primarily working-memory capacity that determines task success (Ackerman, 1986). A second explanation is that procedural learning was defined in this study quite narrowly: as learning orthogonal to associative learning.

[7] However, we computed two new variables: "first" and "last" -- sums of the time spent in the first and last problems of each different problem type. We hypothesized that the correlation between WM and "first" would be higher than with "last," while the correlation between PS and "last" would be higher than with "first." These hypotheses were not supported. For WM, the correlations to first and last were: .51 and .47. For PS, the correlations to first and last were: .38 and .41.

We found that associative (declarative) and procedural learning are distinct factors. First, to account for learning on the tutor itself, it was necessary to posit a procedural learning factor in addition to the general associative learning factor. Removing the procedural learning factor from the model resulted in significantly poorer fits for the model. Moreover, the procedural learning indicators had significant factor loadings on the procedural as well as on the general (declarative) learning factors. Second, the two factors were predicted by different cognitive variables. Incoming knowledge predicted declarative learning, but was unrelated to procedural learning. Third, the two factors influenced transfer in different ways. Only declarative learning predicted transfer.

Certain learning behaviors were also shown to be differentially important at various stages of cognitive skill acquisition. For instance, being "busy" (i.e., number of actions per minute) early in the tutor had a positive relationship to transfer. Being busy later in the tutor was negatively related to transfer. Anderson (1987) similarly found a negative "busy" function while investigating transfer among different text editors. While the actual time per keystroke in the execution of a text-editing procedure did not decrease over the course of the experiment, there was a reduction in the number of keystrokes made per edit. This reflected the acquisition of more efficient procedures over time. The busy index discussed in this paper showed the same pattern of attenuation over time associated with more efficient procedures (or acquisition of the cognitive skill).

In summary, findings showed that a high proportion (86%) of variance in a factor governing the acquisition of a new complex cognitive skill (i.e., Pascal programming) could be accounted for by just three factors: working-memory capacity, domain-related knowledge, and efficiency of acquiring declarative knowledge. Because so much of the learning variance is explained by incoming knowledge and cognitive abilities, as well as by certain learning behaviors, this information can be used in designing ITSs to really enhance learning. For instance, the tutor could provide supplemental instruction (comparable to the Pretutor's declarative knowledge instruction) to individuals with less incoming domain-specific knowledge. Or the system could be programmed to encourage various learning behaviors found to be important at different stages of learning (i.e., hints and runs). To illustrate, freedom to request unlimited hints (as was possible in this ITS) yielded poorer transfer performance as learners failed to work out solutions themselves. But hint-asking seemed not as detrimental in the early compared to the later stages of learning. Tutoring systems could employ some heuristic to wean subjects off the hint option if they tended to be overusing it. Along the same lines, subjects not running the more complex programs but who experienced difficulties in the tutor, could be encouraged to run the programs and see the flow of control. Finally, the implications of the "busy" findings for ITS design include having the system encourage activities in the early stages of the tutor, and support more focused activities later on. It remains an empirical question whether learning behaviors can, in fact, be instructed.[8] Does progressively reducing the number of hints subjects may receive from an ITS significantly increase transfer task performance? Do suggestions to run complex programs affect learning? Are they even heeded? Our facilities provide an easy test for these questions.

Although ITSs are usually created to serve as instructional systems, they are extremely well suited to investigate cognitive learning principles and instructional strategies (Anderson, 1983; Kyllonen & Christal, 1989; Recker & Pirolli, 1990; Shute, 1990; Woolf & Cunningham, 1987). ITS programs can provide detailed traces of learning performance, states of knowledge, and rates of progress through the curriculum. Through these controlled yet rich research vehicles, we can continue to test psychological and instructional theories that could not have been tested before.

[8] Note that Shute and Glaser (1990) did investigate teaching learning behaviors (i.e., scientific inquiry skills) via an ITS. Findings implied that behaviors can be instructed if they can be specified into rules and presented at the appropriate times.

REFERENCES

Ackerman, P.L. (1986). Individual differences in information processing: An investigation of intellectual abilities and task performance during practice. *Intelligence, 10*, 101-139.

Ackerman, P.L. (1988). Determinants of individual differences during skill acquisition: Cognitive abilities and information processing. *Journal of Experimental Psychology: General, 117*, 288-318.

Anderson, J.R. (1983). *The architecture of cognition.* Cambridge, MA: Harvard University Press.

Anderson, J.R. (1987). Skill acquisition: Compilation of weak-method problem solutions. *Psychological Review, 94*(2), 192-210.

Anderson, J.R. (1990). Analysis of student performance with the LISP tutor. In N. Fredericksen, R. Glaser, A. Lesgold, & M. Shafto (Eds.), *Diagnostic monitoring of skill and knowledge acquisition.* Hillsdale, NJ: Lawrence Erlbaum Associates.

Bentler, P.M. (1989). *EQS: Structural equations program manual.* Los Angeles: BMDP Statistical Software, Inc.

Bentler, P.M., & Weeks, D.G. (1980). Linear structural equations with latent variables. *Psychometrika, 45*, 289-308.

Bonar, J., Cunningham, R., Beatty, P., & Weil, W. (1988). *Bridge: Intelligent tutoring system with intermediate representations* (Technical Report). Pittsburgh, PA: Learning Research and Development Center, University of Pittsburgh.

Brooks, R. (1977). Toward a theory of the cognitive processes in computer programming. *International Journal of Man-Machine Studies, 9*, 737-751.

Chi, M.T.H., Glaser, R., & Rees, E. (1982). Expertise in problem solving. In R. Sternberg (Ed.), *Advances in the psychology of human intelligence (Vol. 1).* Hillsdale, NJ: Lawrence Erlbaum Associates.

Cronbach, L.J., & Snow, R.E. (1977). *Aptitudes and instructional methods: A handbook for research on interactions.* New York: Irvington.

Jeffries, R., Turner, A.A., Polson, P.G., & Atwood, M.E. (1981). The processes involved in designing software. In J. R. Anderson (Ed.), *Cognitive skills and their acquisition.* Hillsdale, NJ: Lawrence Erlbaum Associates.

Joreskog, K.G., & Sorbom, D. (1988). *LISREL 7, A guide to the program and applications.* Chicago: SPSS.

Kanfer, R., & Ackerman, P.L. (1989). Dynamics of skill acquisition: Building a bridge between abilities and motivation. In R.J. Sternberg (Ed.), *Advances in the psychology of human intelligence (Vol. 5).* Hillsdale, NJ: Lawrence Erlbaum Associates.

Kyllonen, P.C., & Christal, R.E. (1989). Cognitive modeling of learning abilities: A status report of LAMP. In R. Dillon & J. W. Pellegrino (Eds.), *Testing: Theoretical and applied issues.* San Francisco: Freeman.

Kyllonen, P.C., & Christal, R.E. (1990). Cognitive abilities measurement battery, Version 1. Unpublished computer program.

Kyllonen, P.C., & Christal, R.E. (in press). Reasoning ability is little more than working-memory capacity?! *Intelligence, 14.*

Kyllonen, P.C., & Shute, V.J. (1989). A taxonomy of learning skills. In P.L Ackerman, R.J. Sternberg, & R. Glaser (Eds.), *Learning and individual differences*. New York: Freeman.

Kyllonen, P.C., & Stephens, D. (in press). Cognitive abilities and learning logic gates. In *Learning and individual differences*.

Kyllonen, P.C., & Woltz, D.J. (1989). Role of cognitive factors in the acquisition of cognitive skill. In R. Kanfer, P.L Ackerman, & R. Cudeck (Eds.), *Abilities, motivation, and methodology*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Mayer, R. E., Dyck, J.L., & Vilberg, W. (1986). Learning to program and learning to think: What's the connection? *Communications of the ACM*, 29(7), 605-610.

Melton, A.W. (1967). Individual differences and theoretical process variables: General comments on the conference. In R.M. Gagne (Ed.), *Learning and individual differences*. Columbus, OH: Merrill.

Pellegrino, J. W. (1988). *Individual differences in skill acquisition: Information processing efficiency and the development of automaticity* (AFHRL-TP-87-52, AD-A198 310). Brooks AFB, TX: Manpower and Personnel Division, Air Force Human Resources Laboratory.

Pena, M.C. (1989). *Cognitive determinants of the early stages of programming skill acquisition*. Unpublished master's thesis, St. Mary's University, San Antonio, TX.

Recker, M.M., & Pirolli, P. (1990, April). *A model of self-explanation strategies of instructional text and examples in the acquisition of programing skills*. Paper presented at the American Educational Research Association (AERA), Boston, MA.

Schmalhofer, F. (1982). Comprehension of a technical text as a function of expertise. Doctoral dissertation, University of Colorado. *Dissertation Abstracts International, 44*, 293B.

Shute, V.J. (1990, April). *A comparison of two computer-based learning environments: Which is better for whom and why?* Paper presented at the American Educational Research Association (AERA), Boston, MA.

Shute, V.J., & Glaser, R. (1990). *Large-scale evaluation of an intelligent tutoring system: Smithtown. Interactive Learning Environments, 1*, 51-76.

Snow, R.E., Kyllonen, P.C. & Marshalek, B. (1984). The topography of learning and ability correlations. In R.J. Sternberg (Ed.), *Advances in the psychology of human intelligence (Vol 2)*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Soloway, E., Bonar, J., & Ehrlich, K. (1983, November). Cognitive strategies and looping constructions: An empirical study. *Communications of the ACM, 26.*

Spohrer, J., Soloway, E., & Pope, E. (1985). A goal/plan analysis of buggy Pascal programs. *Human-Computer Interaction, 1.*

Thurstone, L.L. (1938). Primary mental abilities. *Psychometric monographs, No. 1*.

Thurstone, L.L. (1947). *Multiple factor analysis*. Chicago: University of Chicago Press.

Walker, C.H. (1987). Relative importance of domain knowledge and overall aptitude on acquisition of domain-related information. *Cognition and Instruction*, 4(1), 25-42.

Woltz, D. J. (1988). An investigation of the role of working memory in procedural skill acquisition. Journal of *Experimental Psychology: General, 117*, 319-331.

Woolf, B., & Cunningham, P. (1987, Summer). Multiple knowledge sources in intelligent tutoring system, *IEEE Expert*, pp. 41-53.

APPENDIX: BATTERY OF CAM-1 TESTS

I. WORKING MEMORY (Percent correct)

A. Quantitative

1. ABC Recall: Subjects must learn and remember numeric values assigned to the letters A, B, and C. Statements (e.g., A = B/2) are presented one at a time, and subjects are permitted to look at each one for as long as desired before going on to the next statement. They are then asked to recall the values of the letters one at a time. Some of the problems are more difficult than others since values must be computed (e.g., A = 2 x 8 or, A = 16). Still other values cannot be computed until the value of another letter is known (e.g., B = A + 4). Even-odd reliability = .95.

2. Mental Math: This task requires subjects to calculate a subtraction or division problem mentally, and then choose the correct answer from 5 alternatives. A problem appears on the screen for 2 seconds (preceded by a warning asterisk) and then disappears. Subjects mentally solve the problem for as long as they wish. When they have the answer, they hit the space bar to see the five alternatives. They have 4 seconds to type in the number of the correct answer. Even-odd reliability = .88.

3. Slots Test: This test presents simple math equations (e.g., 5 + 2) in five sequential positions on the screen. Subjects must calculate the equations as they are presented and remember the answer for each position. Following the presentation of all equations, a question mark appears in one of the positions, and subjects must type in the corresponding answer. The five positions are marked by horizontal lines, one next to the other. Problems are presented from left to right, one at a time. Two rates of presentation exist (i.e., slow and fast) and before each trial, subjects are warned to get ready for either a slow or fast item. Each problem presents between 1 and 10 math equations. In the more difficult items, new math problems may be presented in a slot where a problem was already presented. Subjects are required to remember the most recent answer. Even-odd reliability = .91.

B. Verbal

4. ABCD Test: Subjects are presented with five rules:
Rule 1 - Set 1 = A and B.
Rule 2 - Set 2 = C and D.
Rule 3 - Set 1 can either precede or follow Set 2.
Rule 4 - A can either precede or follow B.
Rule 5 - C can either precede or follow D.

Each problem consists of three instructions presented one at a time concerning Sets 1 and 2. For example: 1) A precedes B. 2) Set 1 follows Set 2. 3) C precedes D. Each instruction is presented one at a time, and subjects may look at each one as long as desired before going on to the next one. They must determine the appropriate order of the letters and then hit the space bar to see the choices of answers. They then choose the number for the correct answer (e.g., CDAB). Even-odd reliability = .81.

5. Word Span Test: Subjects are required to memorize a short list of words and answer questions about them. A "Get Ready!!" warning precedes the words, which are presented one at a time. The questions are asked in an equation-like format. For example, if the list were 'neat, burp, inn', a possible question is 'neat + 1 = ?'. This question asks for the word which is one position after 'neat'. Answers are presented in a multiple choice format; alternatives are synonyms to the actual words on the list. Subjects must type in the number for the synonym which matches the word from the list. Any given word list is between 3 to 5 words long. Subjects answer three questions about each list, after which they are told how many questions they had correct for that word list. Even-odd reliability = .93.

6. Reading Span: This task tests subjects' ability to classify true/false statements and their short-term memory capacity. Subjects are presented a list of sentences of general knowledge which they must determine to be true/like ('L') or false/different ('D'). Concurrently,

they must memorize the last word in each sentence (this word is highlighted a different color from the other words). Sentences are presenteJ one at a time, after which they are asked to type in the first two letters of each word in the order that they appeared. Subjects receive partial credit if the correct letters are typed in, but in the wrong sequence. Even-odd reliability = .93.

C. Spatial

7. Figure Synthesis: Two geometric figures are presented for subjects who are instructed to imagine the shape if the pieces were rearranged to form one figure. These figures are then replaced by a third figure. The subject must determine whether or not the third figure could be formed from the combined figures. Reaction time is presented when subjects give the correct response. Even-odd reliability = .65.

8. Spatial Visualization: This task requires 3-dimensional visualization. Subjects read descriptions of blocks and visualize how they appear before and after various manipulations (e.g., colors, initial size, ensuing size, number of blocks it may be cut into, etc.). The subject is allowed to study the description for 30 seconds before the first question is asked (although the description remains on the screen throughout the problem). Subjects work the problems mentally and then choose one of the multiple choice answers using the letters A through O. Subjects are given 60 seconds to respond, at which time they are told to enter their response (within another 10 seconds). If no response is entered during that time, the item is counted wrong. For each description, three or more questions may be asked in this multiple choice format. Even-odd reliability = .84.

9. Ichikawa: This test presents a 5x5 matrix of squares containing 7 asterisks. The placement of the asterisks is random. Subjects see a warning asterisk, the matrix filled with asterisks, and then a blank matrix with a question mark in one of the squares. Subjects are to determine whether or not an asterisk was in that square, and respond with 'L' (correct) or 'D' (not correct). Subjects have 3 seconds to respond, and then a new blank matrix appears with another question mark in it. For each matrix, three positions are questioned. The computer provides accuracy feedback. Subjects are allowed to study the initial matrix for 2 seconds, followed by a 1 second delay before questions are asked. Even-odd reliability = .73.

II. INFORMATION PROCESSING SPEED (Latencies)

A. Quantitative

10. Number Fact Reduction: Subjects are given four sets of simple arithmetic problems, each set containing only one type of operation (i.e., addition, subtraction, multiplication, or division). Each problem is preceded by a "Get Ready!!" warning and asterisk. Subjects must quickly determine whether or not the problem is correct (type in 'L') or incorrect (type in 'D'). The computer gives accuracy feedback. Even-odd reliability = .98.

11. Larger-Smaller Test: Subjects are presented with two single-digit numbers on separate sides of the screen to determine which of the two is larger. If the one on the right is greater, 'L' is the correct response; and if the one on the left is greater, 'D' is the correct response. Each set of numbers is preceded by a warning asterisk and a one second delay. The test contains four sets of 36 trials. Even-odd reliability = .98.

12. Odd-Even Test: In this test subjects must decide as quickly as possible whether two numbers presented are odd or even. The two numbers (between 1 and 20) are presented one above the other. Some numbers are presented as digits and others as English words (e.g., 5 or five). Subjects respond with 'L' if both numbers are either odd or even. If one number is odd and the other is even, then 'D' is the correct response. Reaction time is shown when a response is entered. Even-odd reliability = .97.

B. Verbal

13. Meaning Identity: Two words are presented and the subject must decide whether they have the same or different meanings. Subjects type in 'L' if they have the same meaning,

and 'D' if they have different meanings. Some of the pairs of words are repeated exactly and some are repeated with different pairings. Each pair is preceded by a warning asterisk. The goal is to respond as quickly as possible and still try to get 95% of the items correct. After each set, the student's percent correct and average response time are shown. Even-odd reliability = .98.

14. Category Identification: Subjects are presented with three words: one on the left, one on the right, and one centered above the other two. The subjects must determine which of the lower words belongs in the same class or category as the word at the top. If it is the one on the left, 'D' is the correct response, and 'L' if it is the one on the right. Three warning asterisks are presented where the words will appear for the subjects to focus their attention. The computer responds with whether their answer is correct or incorrect in addition to reporting their reaction time. Even-odd reliability = .98.

15. Semantic Relations Verification Test: Subjects must determine whether or not simple sentences are true ('L') or false ('D'). Key words in the sentence are colored. For example, in the sentence 'Theft is a crime', 'theft' and 'crime' might be colored differently from the default color of white like the rest of the sentence. The computer responds with whether or not the subject made the correct response, and the reaction time if the response is correct. Even-odd reliability = .98.

C. Spatial

16. Santa's Figures: Subjects are presented with 2 sets of geometric figures to determine if they have the same parts ('L') or different parts ('D'). Each set consists of three geometric shapes or figures (e.g., circle, arrow, diamond, square) next to each other. The first set appears for 2 seconds, disappears, and then the second set appears. Subjects then make their decisions. The order of the figures is not important, only whether or not the two sets contain the same figures. Even-odd reliability = .93.

17. Palmer's Figure Comparison: Subjects are presented with two 3x3 dot-matrices with 5 interconnecting lines forming different geometric patterns. These matrices are presented side by side until a response is entered. The subject must respond as quickly as possible whether the two line figures are the same ('L') or different ('D'). The computer responds with either 'correct' or 'wrong'. At the end of each set the computer provides summary performance feedback. Even-odd reliability = .96.

18. String Matching: This task presents two strings of letters, symbols, or digits for subjects to determine if they are the same ('L') or different ('D'). The strings may be upper or lower case letters, numbers, or any other symbol from the keyboard (e.g., * & % $). Each string is between 2 to 5 characters long. The two stimuli comprising a comparison are always the same length and composed of the same character type (e.g., upper case with upper case, symbols with symbols, etc.). Strings may be made up of the exact characters, but if the sequence is different, then it is not a match. In all cases, the two strings will either be identical, transposed, or just one value may be different. Subjects are presented a warning asterisk and then the two strings. The strings stay on the screen until a response is entered. At that time, the computer responds with whether the response was correct and the reaction time. Even-odd reliability = .97.

III. GENERAL KNOWLEDGE (Percent correct)

A. Verbal

19. General Knowledge Survey: Subjects are asked general questions (e.g., San Antonio is in what state?), and must respond by typing in the first two letters of the answer (e.g., 'TE' for Texas). The computer responds with 'correct' or 'wrong'. Even-odd reliability = .93.

20. Reading Span True/False: This task tests subjects' abilities to classify statements as either true or false. Subjects are presented a list of sentences of general knowledge which they must determine to be true/like ('L') or false/different ('D'). Even-odd reliability = .72.

21. Meaning Identity: Two words are presented and the subject must decide whether they have the same or different meanings. Subjects are to type in 'L' if they have the same meaning, and 'D' if they have different meanings. Even-odd reliability = .80.

22. Semantic Relations Verification Test: Subjects must determine whether or not simple sentences are true ('L') or false ('D') (e.g., 'Theft is a crime' would be a true sentence). The computer responds with whether or not the subject made the correct response, and the reaction time if the response is correct. Even-odd reliability = .84.

23. General Science score from the ASVAB: This test asks basic questions about the biological and physical sciences. For example, "A rose is a kind of ." Of the four alternatives (i.e., animal, bird, flower, fish), "flower" best answers this question. Subjects are allowed 11 minutes to answer the 25 items in this subtest. Even-odd reliability = .81.

24. Word Knowledge score from the ASVAB: Items consist of a sentence or a phrase with a word or phrase underlined (e.g., It was a small table). Subjects are to determine which of the four lettered alternatives has the same/similar meaning (e.g., a. sturdy, b. round, c. little, d. cheap). This subtest was developed to assess subjects' knowledge of commonly used words. Subjects are allowed 11 minutes to answer the 35 items in this subtest. Even-odd reliability = .89.

III. SPECIFIC KNOWLEDGE (Percent correct)

25. Test 1 -- Examples: There are ten concepts investigated (e.g., integer, string, variable). In this test, the subject has to decide whether an item is or is not an example of a particular concept (e.g., Is 9.7 an example of an integer?) Each of the 10 concepts have 8 corresponding items for a total of 80 items on this test. Half of the items are examples and half are not.

26. Test 2 -- Definitions: The second test requires the subject to decide whether a statement about a concept is true or false (e.g., A variable can only have numeric values). There are 4 related questions per concept for a total of 40 test items (half true, half false)

27. Test 3 -- Bugs: The last test presents a problem statement, then a corresponding program solution (in Pascal code), and a menu of possible error types, or 'Program OK'. The subject indicates the type of error present in the code, if applicable, then must draw a line in the program indicating the faulty or missing line of code. There are five possible error types: 1 line missing, 2 lines missing, error in line, misplaced line, or unnecessary line. Five of these problems are given.